Procedia Computer Science

Volume 51, 2015, Pages 2729–2733

ICCS 2015 International Conference On Computational Science

# The Framework for Rapid Graphics Application Development: The Multi-scale Problem Visualization

Alexey Bezgodov, Andrey Karsakov,
Aleksandr Zagarskikh and Vladislav Karbovskii
*ITMO University, Saint-Petersburg, Russia*
demiurghg@gmail.com, kapc3d@gmail.com, alazar.az@gmail.com,
vladislav.k.work@gmail.com

**Abstract**
Interactive real-time visualization plays a significant role in simulation research domain. Multi-scale problems are in need of high performance visualization with good quality and the same could be said about other problem domains, e.g. big data analysis, physics simulation, etc. The state of the art shows that a universal tool for solving such problem is non-existent. Modern computer graphics requires enormous efforts to implement efficient algorithms on modern GPUs and GAPIs. In the first part of our paper we introduce a framework for rapid graphics application development and its extensions for multi-scale problem visualization. In the second part of the paper we provide a prototype of multi-scale problem's solution in simulation and monitoring of high-precision agent movements starting from behavioral patterns in an airport and up to world-wide flight traffic. Finally we summarize our results and speculate about future investigations.

*Keywords:* Visualization, Multi-scale, GIS, Virtual reality, Software development

## 1 Introduction

Visualization has become an important tool for scientific applications, such as various simulation systems [1, 2] or serious games [3]. However, the current state of the scientific simulations is characterized by scattered research projects using a variety of models or entire systems that have their own visualization solutions created from the scratch [4] or on the basis of the variety of existing tools. The motivation behind this study is prior experience of our research institute in a wide range of simulations [5, 6, 7] and high performance computing [8]. Over the past decade real-time visualization domain underwent a drastically breakthrough in performance and quality level. A substantial contribution to its development has been made by the game industry [9]. Growing demand and fast development of this industry stimulated the emergence of a multitude of software solutions such as game engines. The recent years were marked by the prevailing tendency to use game engines as visualizations tools, in contrast to the development of such tools from scratch [10].

Selection and peer-review under responsibility of the Scientific Programme Committee of ICCS 2015



doi:10.1016/j.procs.2015.05.406

Most of the existing commercial game engines (CryEngine, Unity3d [11], Visual3d etc.) have a sufficiently high cost and even a purchase of a full license does not guarantee the full access to all the source code. On the other hand, there are some cheap or shareware versions of commercial software [12], but with significant limitations in functionality. Unreal Engine 4 provides a full access to all the source code, but it is a huge complicated system and we need to keep in mind that game engines with their redundant diversity of tools are designed to create games, not for scientific purposes.

If we will look at the platforms like XNA Game Studio or MonoGame, they have a convenient development environment, but have a significant disadvantages: XNA is based on outdated DirectX9 (DX9) and its support had been discontinued; MonoGame supports an actual APIs, but developers get access only to a feature list of DX9 in order to support multiplatform devices with slow hardware.

Separately, we want to highlight a GIS features' support as an essential part of a wide range of scientific visualization. Unfortunately, game engines and game development platforms at best are limited in GIS functions that is forced to carry out the preliminary preparation of geo data to be used within the application. We can draw attention to the powerful geo information systems, such as ArcGIS, but many of them are expensive and limited to work with geo data only. Also these systems do not allow developers to create a multi-domain applications on its basis or show a poor performance since they were created on non-optimal development platform.

In this paper we propose project_Fusion, an open source/open architecture framework for rapid prototyping of visualization and interaction applications. The main goal of this framework is to facilitate fast development of software with high visual quality and high performance, as well as to provide unified development environment for multiple research groups. To address these issues, we focused on providing support of modern graphic API, easy-to-use class-based services and defined API to interface various simulation systems.

# 2 Project_Fusion

The overall system architecture of a project_Fusion framework is presented in Figure 1. General architecture is close to the structures of the well-known game engines and game development platforms and is built in layers. Project_Fusion is developed on the basis of .NET Framework 4.5 that provides a built-in runtime platform and using C# as the main programming language.

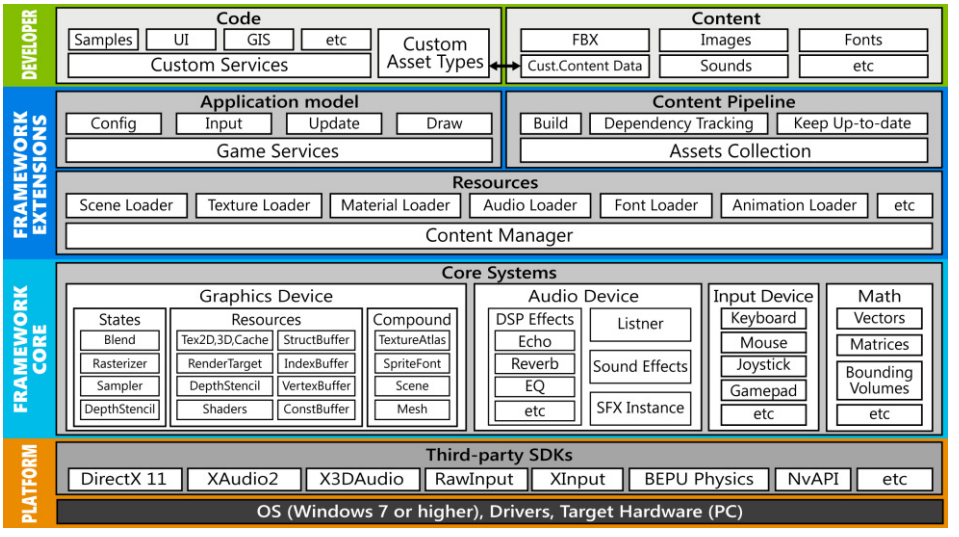


**Figure 1.** Architecture of a project_Fusion framework

The lowest layer, in fact, does not belong to the framework and represents only its underlying platform. As the most game engines [13] and development platforms, project_Fusion framework also leverages a number of third-party SDKs and middleware. Core Framework represents a class library for math and a system-related API. Math library is inherited from SharpDX library with minor changes and extensions (MIT license allows this). Core framework System API is divided into graphics, audio and input devices. It is designed to reduce the amount of work for typical use-cases for each device and to eliminate possible erroneous situations at compile-time and runtime stages.

Application model delivers the main structure of interactive application and includes base classes: GameService is a base class for each application subsystem and Game is a base class for an entire application that aggregates subsystems and designates their execution. Content management system requires flexibility, performance, reduction of disc space occupied by content. Main entity of Content Pipeline in the framework is an Asset - a named object that describes how to build a content entry. AssetCollection provides building, dependency tracking and keeping target content up-to-date.

The basic functionality of the framework was extended by built-in services: GIS and User Interface services. GIS service provides access to the visualization of maps within an application with dynamic load of tile maps from online services. Visualization of maps is possible on a plane with top view as well as in perspective on a virtual globe. The service provides a layered-based approach to working with geo data: base map layer as a basis and any number of custom layers for visualization of developer's information using a range of built-in tools. A part of the calculations that carried out for rendering is transferred to the GPU that significantly increases the performance of the software.

Another built-in service allows to create a graphical user interface (GUI) in a short time for interacting with the application. GUI service allows to create different layouts (with independent areas), as well as the basic elements of the user interface such as panels, buttons, etc.

# 3 Implementation Example

To verify the proposed approach to create visualization and interaction software for various simulation systems the integration with PULSE environment and separate population dynamics model were chosen. PULSE is intended to construct models of urban and socio-economic processes. The agent-based models of different scales were used as the scenario: micro level – public buildings and places; meso level – town blocks or entire city [14]. As a part of integration we used a client-server architecture with asynchronous interaction. Communication protocol is implemented on the basis of http-requests. The following main groups of queries might be distinguished: (a) scenario launch and (b) data exchange in JSON and binary formats.

As a test-bed we have created multi-scale visualization software to demonstrate the results from diversity of simulation scenarios in Pulkovo Airport in Saint-Petersburg. Multi-scale simulations were provided by the several level models within a single scenario - from population dynamics between municipalities within the metropolitan area to the behavior of individuals inside the building.

For representation of macro-level data, visualization of airplane routes from Pulkovo Airport and visualization of population dynamics in Saint-Petersburg were developed. Visualization of spatial graph of airplane routes is presented in Figure 2 (left). A real-time publicly available flights schedule from official website of Pulkovo airport was used as sufficient input data for that visualization - geolocation for graph lines and time schedule for flights dynamics. By zooming in on the Saint-Petersburg area we can see the population dynamics simulation results (Figure 2, right). As input data system takes coordinates of municipal districts, mobility graph (lines of variable width between the centers of districts) and normalized values of population dynamics (polygons' colors from blue to red - the maximum population decline and growth, respectively) for each municipal district.

Meso level is used to visualize scenarios on the scale of the town or city district and it is implemented as follows. For meso and micro level simplified and detailed 3D model of the airport was

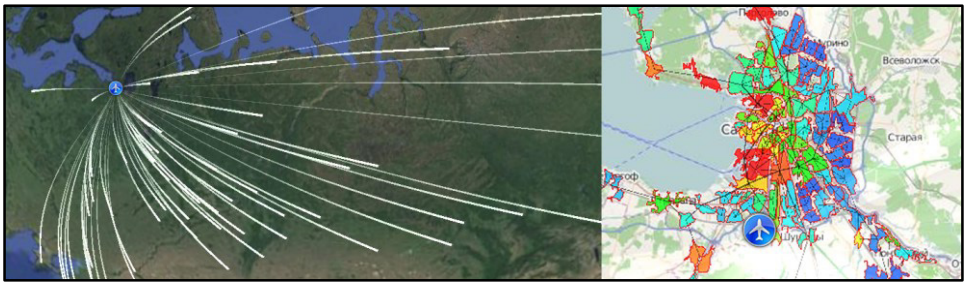


**Figure 2.** Macro scale visualization: left – a virtual globe with spatial graph of flights; right – results of population dynamic simulation in Saint-Petersburg

created in Autodesk 3ds max. By zooming in the map to the airport area a simplified 3D model that is placed on a map can be seen (Figure 3, left). Transition between the meso and micro levels is implemented by mixing render targets of GIS visualization and detailed visualization at micro level. When a transition between levels is required, micro visualization receives a view angle and position of the camera as initial data, so the transition looks very smoth and almost seamless – Figure 3 (from left to middle).

To visualize the airport on the micro level a detailed 3D model was used. Also the ability to switch shading types of the building between solid (Figure 3, middle) and translucent view (Figure 3, right) has been implemented. Agents inside the building are depicted as colored (according to the agents' states) cylinders. Their coordinates are obtained from the simulation server in real-time. GUI consists of a legend, current flights' schedule and a bottom panel with display settings buttons. In addition, the micro-level visualization was used to demonstrate the results of the alignment and optimization of sensor locations (e.g. metal detectors). Expanding the functionality costs a minimum of effort: create 3d models of detectors, algorithm that displays the dynamics of optimization process and some additional elements in GUI.

# 4 Discussion and Conclusions

Studying the above example, its application is plainly the territorial multi-scale. Though we are determined that implementation of our framework might be equally effective for multi-scale examples in other domains, e.g. body-organ-cell. Accuracy in calculations on the GPU still appears to be another topical issue for discussion and an area of future work. And now it is one of the major obstacle to an absolutely seamless and imperceptible transition between the levels of visualization.

In this paper we presented our approach to the development of framework for rapid prototyping of visualization software as well as its application to the development of multi-scale visualization system for PULSE simulation framework on the basis of various scenarios in the Pulkovo airport in St.

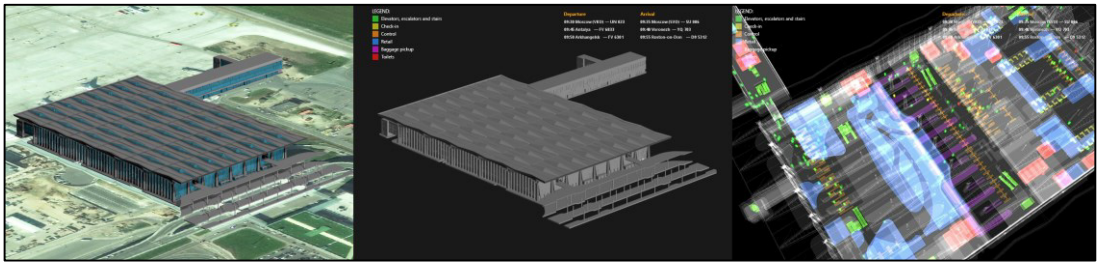

**Figure 3.** Mesoscale visualization of (left) an airport model on the map, (middle) micro scale visualization of the airport after transition from mesoscale and (right) micro scale visualization of airport with X-Ray shading.

Petersburg, Russia. The method of integration with simulation systems presented in this paper has shown its effectiveness in a real example of visualization of the multi-scale simulation results from different systems within a single application. The presented framework stands out of the crowd of its analogues: it provides full support for modern DX11 API, not overwhelmed by excessive features as the game engines and has a flexible built-in implementation of GIS functions.

Since our framework is open source, its functionality of all built-in services could obviously be improved, expanded or redesigned. Also we would like to emphasize that the framework proposed in this paper is a work in progress. It is intended to be a draft that will be modified according to the feedback we will receive from the broader community after publishing it under open source license.

This paper is supported by Russian Scientific Foundation, grant #14-21-00137 "Supercomputer simulation of critical phenomena in complex social systems". The research is done in Advanced Computing Lab (ITMO University), which is opened in frame of 220 Decree of Russian Government, contract #11.G34.31.0019.

# References

[1] A. Falcone, A. Garro, F. Longo and F. Spadafora, "Simulation Exploration Experience : A Communication System and a 3D Real Time Visualization for a Moon base simulated scenario," *IEEE/ACM 18th International Symposium on Distributed Simulation and Real Time Applications,* pp. 113-120, 2014.

[2] C. Smaragda, M. Despina, A. Gregoriades and M. Pampaka, "Design of a 3D interactive simulator for driver behavior analysis," *Simulation Series,* no. 45, pp. 231-238, 2013.

[3] A. Navarro, J. V. Pradilla and O. Rios, "Open Source 3D Game Engines for Serious Games Modeling," *Modeling and Simulation in Engineering,* pp. 143-158, 2012.

[4] M. I. Pomerantz, A. Jain and S. Myint, "Dspace: real-time 3D visualization system for spacecraft dynamics simulation," *Proceedings - 2009 3rd IEEE International Conference on Space Mission Challenges for Information Technology, SMC-IT 2009,* pp. 237-245, 2009.

[5] V. Krzhizhanovskaya, N. Melnikova, A. Chirkin, S. Ivanov, A. Boukhanovsky and P. Sloot, "Distributed simulation of city inundation by coupled surface and subsurface porous flow for urban flood decision support system," *Procedia Computer Science,* no. 18, pp. 1046-1056, 2013.

[6] A. V. Boukhanovsky and . S. V. Ivanov, "Urgent computing for operational storm surge forecasting in Saint-Petersburg," *Procedia Computer Science,* no. 9, pp. 1704-1712, 2012.

[7] V. Kashirin, "Evolutionary Simulation of Complex Networks Structures with Specific Topological Properties," *Procedia Computer Science,* vol. 29, p. 2401–2411, 2014.

[8] K. Knyazkov, S. Kovalchuk, T. Tchurov, S. Maryin and A. Boukhanovsky, "CLAVIRE: e-Science infrastructure for data-driven computing," *Journal of Computational Science,* vol. 3, no. 6, pp. 504-510, November 2012.

[9] T. Germanchis, W. Cartwright and C. Pettit, "Using Computer Gaming Technology To Explore Human Wayfinding and Navigation Abilities Within a Built Environment," *Computer,* p. 10, 2005.

[10] R. C. Mat, A. R. M. Shariff, A. N. Zulkifli, M. S. M. Rahim and M. H. Mahayudin, "Using game engine for 3D terrain visualisation of GIS data: A review," *IOP Conference Series: Earth and Environmental Science,* no. 20, pp. 1-12, 2013.

[11] S. Wang, Z. Mao, C. Zeng, H. Gong, S. Li and B. Chen, "A new method of virtual reality based on Unity3D," *18th International Conference on Geoinformatics,* pp. 1-5, 2010.

[12] Y. Zhao, C. Yan and X. Zhou, "The research and development of 3D urban geographic information system with Unity3D," *Geoinformatics, 21st International Conference,* pp. 1-4, 2013.

[13] J. Gregory, Game Engine Architecture, 2nd ed., CRC Press, 2014.

[14] V. Karbovskii, D. Voloshin, K. Puzyreva and A. Zagarskikh, "Personal Decision Support Mobile Service for Extreme Situations," *Procedia Computer Science,* no. 29, p. 1646–1655, 2014.